2011 International Conference on Software and Computer Applications
IPCSIT vol.9 (2011) © (2011) IACSIT Press, Singapore

An Integrated Voting Algorithm for Fault Tolerant Systems

Soureh Latif-Shabgahi[1]

[1]Department of Information Studies, The University of Sheffield, UK

Abstract. Fault masking is one of the primary approaches to improve or maintain the normal behaviour of a range of safety-critical systems. Some industrial sectors which employ such systems include process control, transportation, nuclear power station and military applications. Majority and weighted average voters have been widely used in these applications to provide error/fault-masking capability. The former gives a good level of safety and the latter offers a good level of availability. This paper introduces a novel voter that integrates the majority and standard weighted average voters. It is an appropriate substitute for the weighted average voter. An experimental framework for evaluating software voting algorithms is provided and safety/availability behaviour of the proposed voter is studied in various error scenarios. The experimental results show that in all scenarios the integrated voter gives a higher availability than both the majority and weighted average voters. However, it has compromise safety behaviour between the majority and weighted average voters. It also performs faster than the naive weighted average voter does.

Keywords: Dependability, Fault Tolerance, Fault masking, Voting

1. Introduction

Voter is a critical component in the implementation of N-Modular Redundant systems [1,2]. Voting can be a hard problem in itself, for at least three reasons: i) floating point arithmetic is not exact and thus voting on floating point values requires inexact voting, ii) the output of variants (redundant modules) may be extremely sensitive to small variations in critical regions, e.g. around threshold values in the specification, and iii) some problems have multiple correct solutions (e.g. square roots) which may confuse the voter. Different voting strategies have been introduced to handle these problems; examples are maximum likelihood voter [3], predictor voters [4], stepwise negotiation voter [5] and word voters [6]. Two traditional and widely used voting algorithms are the majority and the weighted average voters. In its general form, an inexact majority voter [7] produces a correct output if the majority of its inputs match each other; that is, they are within an application-specific interval of each other. In cases of no majority, the voter generates an exception flag which can be detected by the system supervisor to drive the system towards a safe state. Efficient implementations of the majority voter have been addressed in [8-10].

The weighted average voter on the other hand, calculates the weighted mean of its redundant input values. It is useful in applications such as clock synchronisation in distributed computer systems, pattern recognition and sensor planes where a result has to be generated in each voting cycle. The weights can be predetermined or can be adjusted dynamically. Calculated weights, w_i, are then used to compute the voter output, $z = \Sigma w_i \,.\, x_i \,/\, \Sigma w_i$ where x_i values are the voter inputs and z is the voter output.

The standard majority and weighted average voters are examples of two distinct groups of voting algorithms. One with a high level of safety yet with a low level of availability (the terms 'safety' and 'availability' will be defined later in section 3.3) and the other with a low level of safety yet a high level of availability. This paper introduces a novel voter with a compromise safety performance between the

+ Corresponding author. Tel: 00447946488313.
E-mail address: sourehl@yahoo.co.uk

standard inexact majority and weighted average voters. It also gives a higher availability than both the majority and weighted average voters. The organisation of the paper is as follows. In section 2 the structure of integrated voter is explained. Section 3 gives the structure and characteristics of the experimental framework established. Section 4 compares the safety and availability performance of voters in multiple permanent and transient error injection scenarios. Finally, some conclusions are given in section 5.

2. Integrated Voter

A correctly functioning majority voter with a given consensus-threshold selects at random, from the agreed voter inputs, where a majority exists. A correctly functioning weighted average voter always generates the weighted mean of its inputs that is identical to or in between the inputs that the majority voter would select as in agreement. It is obvious that the output of inexact majority and weighted average voters for all agreement voting cycles are similar. This implication leads us to introduce a novel voter that is a combination of majority and weighted average voters. It performs as a majority voter in agreement cases, and functions as a weighted average voter in disagreement voting cycles. The voter is less complex and quicker than the weighted average voter, since in majority of the cases it does not perform the relatively time consuming weighted averaging procedure. Moreover, the use of the majority algorithm in agreed voting cycles of the novel voter improves its whole safety level compared to the standard weighted average voter.

The novel voter is defined more formally as follows in which the sort and search technique is used in the implementation of majority section (steps 2-4):

1. Let $A = \{x_1\ x_2\ x_3\ \ldots x_n\}$ denote the set of n voter inputs;
2. Sort the set A in ascending order to construct the new set $AS = \{y_1\ y_2 \ldots y_n\}$;
3. Construct the partitions $V_j = \{ y_j\ \ y_{j+1} \ldots y_{j+m-1}\ \}$ from set AS , for all $j = 1: m$, where $m = (n+1)/2$;
4. If at least one of the subsets V_j $(j = 1, 2, \ldots m)$ meets the property $d(y_j , y_{j+m-1}) \leq \epsilon$ then the majority for the original set A will be satisfied. Here ϵ is the consensus threshold and d is a distance metric between elements. Then select the mid-located module result in the set AS as voter output, $z = x([n+1]/2)$;
5. In cases that none of the subsets V_j $(j = 1, 2, \ldots m)$ meets the property $d(y_j , \ldots y_{j+m-1}) \leq \epsilon$, let the weightings w_1 , w_2 , w_N be non-negative real numbers with $s = \sum_{i=0}^{N} wi$, where s is a quantity used to normalise the weightings, so that they sum to a value of one;
6. Compute weightings as dynamic functions of the voter inputs as follows where a is a scaling factor.

$$w_i = \frac{1}{1 + \prod_{\substack{i,j=1 \\ i \neq j}}^{N} \frac{d_{ij}^2}{\alpha^2}}$$

7. Define a voter result as: $z = \sum_{i=1}^{N} \left(\frac{wi}{s}\right) . x\, i$. Thus, the weight assigned to the i[th] voter input, x_i, is inversely proportional to the distance values.

3. Experimental Framework

3.1. Test harness structure and capabilities

The experimental test harness is shown in Figure 1. One notional correct result is produced by the input generator in each test cycle. This sequence of numbers simulates identical correct results generated by redundant variants. Copies of the notional correct result are presented to each saboteur, in every cycle. The saboteurs can be programmed to introduce selected module error amplitudes, according to selected random distributions. It is assumed:

- That the majority voter is inexact. A *consensus threshold*, *a*, is used in such voting algorithms to determine the maximum acceptable divergence of voter inputs from the notional correct result.
- That all voters perform correctly. This assumption is made due to the fact that the voting algorithm is usually a simpler program than the variants (modules) it monitors.

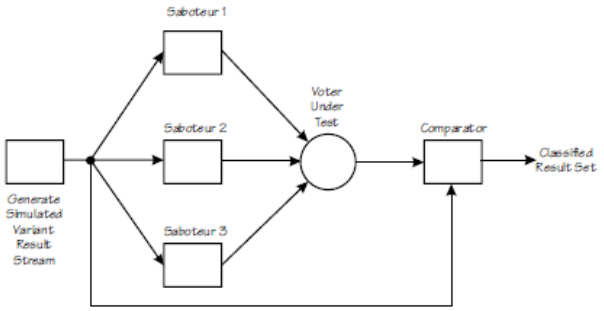

Fig. 1: Experimental test harness.

- That all voters are used in a cyclic system where there exists some relationship between correct results from one cycle to the next (e.g. controlling the fuel supply to an engine);
- That faults cause errors whose symptoms appear to the voter as numerical input values perturbed.
- At any voting cycle the "notional correct result" is known.
- A comparator is used to check for agreement between the notional correct result and the output of the voter under test at any voting cycle.
- An *accuracy threshold*, ϵ , is used, in the comparator, to determine if the distance between the notional correct result and the voter output is within acceptable limits. A voter result which has a distance from the notional correct answer less than the accuracy threshold is taken as a *correct* output, otherwise it is considered as an *incorrect* output.

Within the test harness the following parameters can be adjusted.

- *The value of consensus threshold.*
- *Number of voter inputs.* The test harness provides a facility to define 3, 5 and 7 inputs voters.
- *Input data trajectory and sample rate.* Different types of input data trajectory can be selected within the test harness. The frequency of data arrival (*sample rate* of input data) is also adjustable.
- *The value of accuracy threshold.*
- *The amplitude of injected errors.* The amplitude of injected errors can be expressed as a function of the input signal. If δ is defined as the maximum amplitude of errors during a particular test consisting of n voting cycles, and A is taken as the maximum amplitude of input data, the δ / A will be the *maximum error-to-signal ratio (ESR)* of that particular test.
- *Number of injected errors.* One or two or three saboteurs may be programmed to simulate variant results' errors.
- *Error persistence time.* The experimental harness has capability to select error persistence time, error arrival intervals and its activation period. Errors can be permanent, transient, and intermittent.
- *Error Distribution.* A variety of error distributions including uniform, exponential, normal and Poisson distributions with adjustable parameters have been defined within the test harness.

3.2. Experimental method

A stream of input data with sinusoidal trajectory, and fixed sample rate (0.1 second) feeds both the saboteurs and the comparator. Both the consensus threshold and accuracy threshold parameters are set to the fixed value 0.5. Random errors with uniform distribution from interval $[-\delta +\delta]$ are injected into the saboteurs to simulate permanent (multiple) module errors. The multiple error conditions are selected because there is relatively little published work considering the behaviour of voters in such circumstances. However, such error cases can produce catastrophic consequences and should be considered. The output of the saboteurs is presented to the voter under test. In each voting cycle the output of the voter, z, is compared with a copy of input data, xo. Based on the numerical distance between z and x_o values, the output of the voter is interpreted as *correct, incorrect, or benign* value. For each voter, the results of n system run (n is selected 10^4) are classified. In this way, n_c correct results, n_{ic} incorrect outputs, and n_b benign results are collected. These data are, then, used for evaluation and comparison of selected voters. Two performance measures are defined for this propose:

1. *Safety (S):* The safety measure can be defined as: $S = (1 - n_{ic} / n)$. Thus, $S \in [0\ 1]$ and ideally $S = 1$.

2. *Availability (A)*: The availability of a voter is defined as $A = n_c / n$. Thus, $A \in [0\ 1]$ and ideally $A = 1$
Each of the performance criteria can be measured versus the changes of *error-to-signal ratio*.

4. Experimental Results

4.1. Safety-Availability performance of voters

Figures 2 and 3 indicate the safety and availability plots of majority, standard weighted average, and integrated majority-weighted average voters versus maximum error- to- signal ratio (in percentage) for 10^4 runs of voters for a range of error injection experiments. *ESR* varies from 0.5% to 3% in this set of experiments to examine voters' performance in applications with low-erroneous modules to those with highly erroneous modules. The plots indicate two distinct groups of voting algorithms; one group (majority, here) has a high level of safety yet with a low level of availability and the other (weighted average and integrated weighted average-majority voters, here) offers a low level of safety yet a high level of availability.

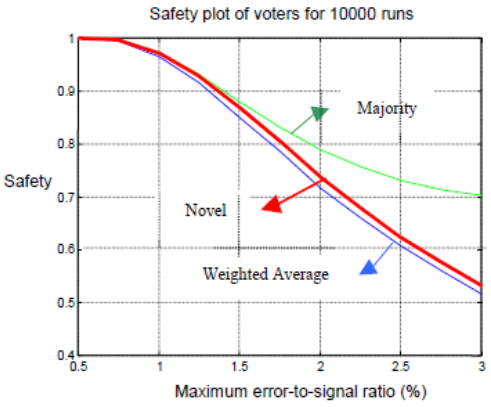

Fig. 2: Safety of voters vs. error-to-signal ratio.

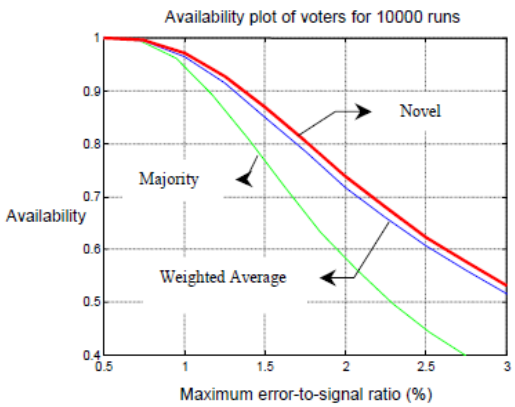

Fig. 3: Availability of voters vs. error-to-signal ratio.

The novel voter shows a compromise safety behaviour between the standard majority and weighted average voters. The plots also show the superiority of the novel voter to the standard weighted average voter in terms of both safety and availability measures; it has at least an improvement of 2% in both safety and availability. At first glance, such an improvement may seem small, however, in high available applications such an improvement (in the presence of permanent errors) is considerable. For small error cases (where $ESR < 1.5$) the safety level of the novel voter is too close to that of the majority voter, however, for larger errors the majority voter still gives the best safety values.

4.2. Safety performance of voters with transient errors

We carried out other experiments in various transient multiple errors to obtain more realistic safety values. In particular, we tested the voters in the following 'error scenarios' for 10^5 runs:

- *Scenario 1.* 1 error is injected in every voting cycle (therefore, 10^5 errors are injected in total)
- *Scenario 2.* 1 error is injected in every *9* voting cycles (therefore, 10^4 errors are injected in total)
- *Scenario 3.* 1 error is injected in every *99* cycles (therefore, 10^3 errors are injected in total)
- *Scenario 4.* 1 error is injected in every *999* cycles (therefore, 100 errors are injected in total)
- *Scenario 5.* 1 error is injected in every 9999 cycles (therefore, 10 errors are injected in total)
- *Scenario 6.* 1 error is injected in every 999,99 voting cycles (therefore, 1 error is injected)

Table 1 indicates the safety value of voters in the presence of transient errors with *ESR=2* with different occurrence frequency. The trend of safety values of each voter can be used to estimate the safety of that voter during longer mission times than reported in this paper. The majority voter, for example, gives $S = 0.7921$ when errors are injected in every voting cycle, $S = 0.9817$ when an error is injected every 9 cycles, $S = 0.99801$ when one error is injected every *99* cycles, $S = 0.999,813$ when one error is injected every 999 cycles, $S = 0.999,981$ when one error is injected every 999,9 cycles and $S = 0.999,998$,when one error is injected every 999,99 cycles. By considering this trend, it is expected that the majority voter will give $S =$

0.999,999,81 if its inputs are perturbed once every 999,999 voting cycles and generated *S = 0.999,999,981 (0.97)* if one error occurs every *999,999,9* cycles and so on.

Table 2. Safety of voters in **triple transient error** scenarios, when *ESR =2*

Voter	Scen. 1	Scen. 2	Scen. 3	Scen. 4	Scen. 5	Scen. 6
Majority	0.7921	0.9817	0.998,01	0.999,813	0.999,981	0.999,9981
Weighted Avg.	0.7231	0.9734	0.997, 26	0.999,723	0.999,972	0.999,9972
Novel	0.7458	0.9747	0.997,51	0.999,750	0.999,975	0.999,9975

5. Conclusions

Fault masking applications usually require a high level of safety and availability, which frequently conflict with each other. The published voters do not offer such capabilities. The majority voter with high level of safety has generally low level of availability and the weighted average voter gives high level of availability in the price of low safety. This paper introduced a voter by integrating the standard majority and distance metric-based weighted average voters and addressed the benefits of this voter. The experimental results showed that the novel voter has a compromise safety performance between the majority and weighted average voters. However, it gives higher availability than both the majority and weighted average voters do. The novel voter offers higher availability and safety than the naive weighted average voter. As a result, wherever the weighted average voter is used, the novel voter would be a better candidate.

6. References

[1] S. Lee, J.-I. Jung, and I. Lee. Voting Structures for Cascaded Triple Modular Redundant Modules, *IEICE Electronics Express*, 2007, **4** (21): 657-664.

[2] A. B. Baykant. Hierarchical Triple-Modular Redundancy (H-TMR) Network For Digital Systems, *OncuBilim Algorithm And Systems Labs.* 2008, **8** (5), URL: http://arxiv.org/ftp/arxiv/papers/0902/0902.0241.pdf

[3] Y. W. Leung. Maximum Likelihood Voting for Fault-Tolerant Software with Finite Output Space", *IEEE Trans. on Reliability*, 1995, **44** (3): 419-427.

[4] J. M. Bass, P. R. Croll, P. J. Fleming, and L. J. C. Woolliscroft. Three Domain Voting in Real-Time Distributed Control Systems", *Proc. of 2nd IEEE Euromicro Workshop on Parallel & Dist. Processing*, 1994, pp. 317-324.

[5] K. Kanekawa, H. Maejima, H. Kato, and H. Ihara. Dependable On-Board Computer Systems with a New Method: Stepwise Negotiated Voting", *Proc. of FTCS'19: IEEE 19th Ann. Int. Symp. on Fault-Tolerant Computing Systems*, 1989, Chicago, USA, pp. 13-19.

[6] S. Mitra and E. J. McCluskey. Design of Redundant Systems Protected Against Common-Mode Failures", *Technical Report, CRC-TR-00-2*, Stanford University, 2000, URL: http://crc.stanford.edu.

[7] P. R. Lorczak, A. K. Caglayan and D. E. Eckhardt. A Theoretical Investigation of Generalised Voters", *Proc. of IEEE 19th Ann. Int. Symp. on Fault-Tolerant Computing Systems,* 1989, Chicago, USA, June, pp. 444-451.

[8] H. Yu and A. Xu. Design of a Fault-Tolerant Voter for Safety Related Analog Inputs", Proc. of 3rd Int. Conf. on Advanced Computer Theory and Engineering, ICACTE, 2010, Chengdu, China.

[9] K. Kwiat, A. Taylor, W. Zwicker, D. Hill, S. Wetzonis, and S. Ren. Analysis of Binary Voting Algorithms for Use in Fault Tolerant and Secure Computing", 2010, URL: http://lcs.syr.edu/faculty/tang/Teaching/CSE791-Spring11/Papers/Voting.pdf

[10] S. Askari, B. Dwivedi, A. Saeed, and M. Nourani. Scalable \Mean Voting Mechanism for Fault Tolerant Analog Circuits", *Proc. of 4th Int. Workshop on Design and Test (IDT),* 2009, Riyadh, Nov. 15-17, pp. 1-6.